I claim:

1. A computing system for decoding a Reed-Solomon-encoded string of data, the computing system comprising a processor circuit operable to:

receive a Reed-Solomon code word; and

decode the code word in software with a constant or approximately constant level of processing.

2. The computing system of claim 1 wherein:

receiving the code word comprises receiving the code word in a time having a duration *t*; and

decoding the code word comprises decoding the code word in a number of steps each having a duration of *t*.

3. The computing system of claim 1 wherein:

receiving the code word comprises receiving the code word in a time having a duration *t*; and

decoding the code word comprises decoding the code word in a number of sequential steps each having a duration of *t*.

4. The computing system of claim 1 wherein:

the code word comprises a number *n* of equal-length blocks;

receiving the code word comprises receiving each of the blocks in a time having a duration *t/n*; and

decoding the code word comprises decoding the code word in a number of sequential steps each having *n* sub steps of a duration *t/n*.

5. A computing system for decoding a Reed-Solomon-encoded string of data, the computing system comprising a processor circuit operable to:

receive a first Reed-Solomon code word;

receive a second Reed-Solomon code word after receiving the first code word; and

while receiving the second code word, decoding the first decoding step of the first code word with a constant or approximately constant level of processing.

6. The computing system of claim 5 wherein:

receiving the second code word comprises receiving the second code word during a time period having a duration *t*; and

decoding the first code word comprises performing a first decoding step of the first code word during the time period.

7. The computing system of claim 5, further comprising:

wherein receiving the second code word comprises receiving the second code word during a first time period having a duration *t*;

wherein decoding the first code word comprises performing a first decoding step on the first code word during the first time period;

receiving a third code word during a second time period having the duration *t*;

performing a second decoding step on the first code word during the second time period; and

performing the first decoding step on the second code word during the second time period.

8. The computing system of claim 5 wherein:

the first and second code words each comprise a number *n* of equal-length blocks;

receiving the first and second code words comprises receiving each of the blocks in *n* sequential time periods each having a duration *t/n*; and

decoding the first code word comprises decoding the first code word in *n* sequential sub steps each having a duration *t/n*.

9. A computing system for decoding a Reed-Solomon-encoded string of data, the computing system comprising a processor circuit operable to:

receive five Reed-Solomon code words, each during a time period having a duration *t*, and each comprising a number *n* of equal-length blocks;

while receiving the second code word, decoding the first step of the first code word in *n* sequential sub steps each having a duration *t/n*;

while receiving the third code word, decoding the second step of the first code word in *n* sequential sub steps each having a duration *t/n* and decoding the first step of the second code word in *n* sequential sub steps each having a duration *t/n*;

while receiving the fourth code word, decoding the third step of the first code word in *n* sequential sub steps each having a duration *t/n* and decoding the second step of the second code word in *n* sequential sub steps each having a duration *t/n* and decoding the first step of the third code word in *n* sequential sub steps each having a duration *t/n*; and

while receiving the fifth code word, decoding the fourth step of the first code word in *n* sequential sub steps each having a duration *t/n* and decoding the third step of the second code word in *n* sequential sub steps each having a duration *t/n* and decoding the second step of the third code word in *n* sequential sub steps each having a duration *t/n* and decoding the first step of the fourth code word in *n* sequential sub steps each having a duration *t/n*.

10. A computing system for decoding a Reed-Solomon-encoded string of data, the computing system comprising a processor circuit operable to:

1. receive the Reed-Solomon-encoded string of data in *t* seconds;
2. calculate a syndrome in said *t* seconds;
3. calculate the coefficients of an error locator polynomial in said *t* seconds;

4. determine the roots of an error locator polynomial in said *t* seconds; and

5. determine the magnitude of an error and correcting the error in said *t* seconds.

11. The computing system of claim 10 wherein said Reed-Solomon-encoded string of data comprises:

a first number of data symbols each including second number of bits; and

a third number of parity symbols each including the second number of bits.

12. The computing system of claim 10 wherein said Reed-Solomon-encoded string of data comprises:

a first number of symbol blocks each including a second number of symbols;

a third number of parity symbols; and

wherein the third number is an integer multiple of the first number.

13. The computing system of claim 10 wherein the processor circuit executes a Berlekamp-Massey Algorithm to calculate the coefficients of the error locator polynomial.

14. The computing system of claim 10 wherein the processor circuit executes a Chien Search to determine the roots of the error locator polynomial.

15. The computing system of claim 10 wherein the processor circuit executes a Forney Algorithm to determine the magnitude of the errors in the received digital code word.

16. The computing system of claim 10, further comprising:

wherein said Reed-Solomon-encoded string of data comprises:

a first number of symbol blocks each including a second number of symbols;

a third number of parity symbols; and

wherein the third number is an integer multiple of the first number; and

wherein the processor circuit is operable to divide each of the tasks (a)-(e) into the first number of subtasks, the processor operable to execute each of the subtasks in a time equal to *t* divided by the first number such that the processor load is even for every symbol block.

17. The computing system of claim 10, further comprising:

wherein said Reed-Solomon-encoded string of data comprises:

a first number of symbol blocks each including a second number of symbols;

a third number of parity symbols; and

wherein the third number is an integer multiple of the first number; and

wherein the processor circuit is operable to divide each of the tasks (a)-(e) into the first number of subtasks, the processor circuit operable to pipeline the tasks (a) – (e) by simultaneously executing one subtask from each task in a time equal to *t* divided by the first number such that the processor-circuit load is even for every symbol block.

18. The computing system of claim 10, wherein:

the processor circuit comprises multiple processors;

said Reed-Solomon-encoded string of data comprises:

a first number of symbol blocks each including a second number of symbols;

a third number of parity symbols; and

wherein the third number is an integer multiple of the first number; and

the processor circuit is operable to divide each of the tasks (a) – (e) into the first number of subtasks, the processor circuit operable to pipeline the tasks (a)-(e) by simultaneously executing one subtask from each task with a respective processor.

19. A method of operating a computing system with a Reed-Solomon decoding application comprising the steps of:

receiving a Reed-Solomon code word; and

decoding the code word in software with a constant or approximately constant level of processing.

20. The method of claim 19 wherein:

receiving the code word comprises receiving the code word in a time having a duration *t*; and

decoding the code word comprises decoding the code word in a number of steps each having a duration of *t*.

21. The method of claim 19 wherein:

receiving the code word comprises receiving the code word in a time having a duration *t*; and

decoding the code word comprises decoding the code word in a number of sequential steps each having a duration of *t*.

22. The method of claim 19 wherein:

the code word comprises a number *n* of equal-length blocks;

receiving the code word comprises receiving each of the blocks in a time having a duration *t/n*; and

decoding the code word comprises decoding the code word in a number of sequential steps each having *n* sub steps of a duration *t/n*.

23. A method of operating a computing system with a Reed-Solomon decoding application comprising the steps of:

receiving a first Reed-Solomon code word;

receiving a second Reed-Solomon code word after receiving the first code word; and

while receiving the second code word, decoding the first decoding step of the first code word with a constant or approximately constant level of processing.

24. The method of claim 23 wherein:

receiving the second code word comprises receiving the second code word during a time period having a duration *t*; and

decoding the first code word comprises performing a first decoding step of the first code word during the time period.

25. The method of claim 23, further comprising:

wherein receiving the second code word comprises receiving the second code word during a first time period having a duration *t*;

wherein decoding the first code word comprises performing a first decoding step on the first code word during the first time period;

receiving a third code word during a second time period having the duration *t*;

performing a second decoding step on the first code word during the second time period; and

performing the first decoding step on the second code word during the second time period.

26. The method of claim 23 wherein:

the first and second code words each comprise a number *n* of equal-length blocks;

receiving the first and second code words comprises receiving each of the blocks in *n* sequential time periods each having a duration *t/n*; and

decoding the first code word comprises decoding the first code word in *n* sequential sub steps each having a duration *t/n*.

27. A method of operating a computing system with a Reed-Solomon decoding application comprising the steps of:

receiving five Reed-Solomon code words, each during a time period having a duration *t*, and each comprising a number *n* of equal-length blocks;

while receiving the second code word, decoding the first step of the first code word in *n* sequential sub steps each having a duration *t/n*;

while receiving the third code word, decoding the second step of the first code word in *n* sequential sub steps each having a duration *t/n* and decoding the first step of the second code word in *n* sequential sub steps each having a duration *t/n*;

while receiving the fourth code word, decoding the third step of the first code word in *n* sequential sub steps each having a duration *t/n* and decoding the second step of the second code word in *n* sequential sub steps each having a duration *t/n* and decoding the first step of the third code word in *n* sequential sub steps each having a duration *t/n*; and

while receiving the fifth code word, decoding the fourth step of the first code word in *n* sequential sub steps each having a duration *t/n* and decoding the third step of the second code word in *n* sequential sub steps each having a duration *t/n* and decoding the second step of the third code word in *n* sequential sub steps each having a duration *t/n* and decoding the first step of the fourth code word in *n* sequential sub steps each having a duration *t/n*.

28. A method of operating a computing system with a Reed-Solomon decoding application comprising the steps:

1. receive the Reed-Solomon-encoded string of data in *t* seconds;

2. calculate a syndrome in said *t* seconds;

3. calculate the coefficients of an error locator polynomial in said *t* seconds;

4. determine the roots of an error locator polynomial in said *t* seconds; and

5. determine the magnitude of an error and correcting the error in said *t* seconds.

29. The method of claim 28 wherein said Reed-Solomon-encoded string of data comprises:

a first number of data symbols each including second number of bits;

a third number of parity symbols each including the second number of bits.

30. The method of claim 28 wherein said Reed-Solomon-encoded string of data comprises:

a first number of symbol blocks each including a second number of symbols;

a third number of parity symbols; and

wherein the third number is an integer multiple of the first number.

31. The method of claim 28 wherein the computing system executes a Berlekamp-Massey Algorithm to calculate the coefficients of the error locator polynomial.

32. The method of claim 28 wherein the processor circuit executes a Chien Search to determine the roots of the error locator polynomial.

33. The method of claim 28 wherein the processor circuit executes a Forney Algorithm to determine the magnitude of the errors in the received digital code word.

34. The method of claim 28, further comprising:

wherein said Reed-Solomon-encoded string of data comprises:

a first number of symbol blocks each including a second number of symbols;

a third number of parity symbols; and

wherein the third number is an integer multiple of the first number; and

wherein the processor circuit is operable to divide each of the tasks (a)-(e) into the first number of subtasks, the processor operable to execute each of the subtasks in a time equal to *t* divided by the first number such that the processor load is even for every symbol block.

35. The method of claim 28, further comprising:

wherein said Reed-Solomon-encoded string of data comprises:

a first number of symbol blocks each including a second number of symbols;

a third number of parity symbols; and

wherein the third number is an integer multiple of the first number; and

wherein the processor circuit is operable to divide each of the tasks (a)-(e) into the first number of subtasks, the processor circuit operable to pipeline the tasks (a) – (e) by simultaneously executing one subtask from each task in a time equal to *t* divided by the first number such that the processor-circuit load is even for every symbol block.

36. The method of claim 28, wherein:

the processor circuit comprises multiple processors;

said Reed-Solomon-encoded string of data comprises:

a first number of symbol blocks each including a second number of symbols;

a third number of parity symbols; and

wherein the third number is an integer multiple of the first number; and

the processor circuit is operable to divide each of the tasks (a) – (e) into the first number of subtasks, the processor circuit operable to pipeline the tasks (a) – (e) by simultaneously executing one subtask from each task with a respective processor.